

第2章 安全性設計の基本を理解する

障害や故障を分析して安全を作り込む

鈴村延保

ここでは，安全性設計を考える上で必要な概念や設計の手順，手法などについて解説する．障害や故障を分析する手法の例としては，自動車分野でよく使われているフォールト・ツリー解析（FTA：fault tree analysis）や故障モード影響解析（FMEA：failure mode and effect analysis）を取り上げ，具体的に説明する．（編集部）

1．安全性設計の概要

安全性や信頼性という言葉は，人によって使い方や受け取り方が異なります．そこで最初に，安全性設計を「潜在危険（hazard）とリスクの低減を狙いとした設計」であると定義したいと思います．潜在危険の一つとして故障（failure）があります．故障は，故障モード（failure mode）の影響がどのようなものであるかによって，どの程度危険であるのかが異なります．

一方，信頼性とは，一般に部品などのハードウェアにおける故障のしにくさを表します．信頼性設計を一言で言うと，「故障の発生を抑えるための設計」であると言えます．例えば，電球のフィラメントが切れるのは，主に経時劣化や偶発故障によるものです．その場合の信頼性設計とは，平均故障間隔（MTBF：mean time between failures）や故障までの平均時間（MTTF：mean time to failure）を，より長くすることを狙いとして設計することです．

● 潜在危険やリスクの少ないソフトウェアが「安全」

それでは，ソフトウェアの安全性と信頼性の定義について考えてみましょう．ソフトウェアの信頼性とは，「ある期間とある条件下において，不具合の発生しない動作が得られる確率」であると言えます．

ソフトウェアの不具合は，劣化や摩耗による故障ではありません．しかし，内在する不具合などによって発生する「故障」に関する信頼性の定義は，前述した（ハードウェアにおける）平均故障間隔（MTBF）のようなイメージでとらえられます．

また，ソフトウェアの「故障」には，メイン・プログラムと割り込みの干渉など，確率的に発生する故障もあれば，ある条件になると必ず発生する故障や，センサや機構などの経時劣化をうまく処理していないために経年で顕在化する故障などがあります．また，不具合としては発生しても，故障にまでは至らないこともあります．

ソフトウェアの安全性設計も，先ほどの定義に準じて，「故障などに起因する潜在危険やリスクの低減を狙いとした設計」であると言えます．

● 欠陥と障害，故障の違いを理解する

安全に関する用語には紛らわしい言葉が出てくるので，用語の関連をまず理解してください．

1）障害（fault）― 異常（error）― 故障（failure）

ある障害（fault）がトリガとなり，異常（error），故障（failure）と連鎖します．障害から故障までの連鎖は，さらに上位の障害に結びつきます．その障害はさらに異常，故

KeyWord
機能安全，IEC 61508，fault，error，failure，hazard，FMEA，故障モード，FTA，HAZOP，JIS C 0508，QS-9000，IEC 16949，ISO 9000，CMM，形式検証，SDL，UML，ADL

障と連鎖します．これを繰り返すと事故(accident)につながります(**図1**)．

2) 欠陥(defect) ― 障害(fault) ― 故障(failure)

ソフトウェアに内在する不具合など，開発時に作り込まれてしまった欠陥(defect)があると，稼働中に障害(fault)が顕在化し，**図1**に示したように故障(failure)へと連鎖します．故障は部品レベルから，システム故障へとつながります．また部品故障は，副次的に第2の部品故障や，それが連鎖して第2のシステム故障へと拡大する場合があります(**図2**)．

● 効果の高い安全性設計はコストもかかる

ここまで述べてきた概念とその関係を理解することにより，どの段階でどのように安全性(安全処置)を設計すれば効果的なのかが見えてきます．安全処置を設計する手順を**図2**の因果関係モデルに沿って整理すると，以下のようになります．

① 欠陥を発生元で予防する
② 欠陥を早い段階で除去する
③ 欠陥を除去できないまでも，被害を低減させる(マージンを取る，防御構造を準備する，など)
④ 顕在化した故障を制御する(フェイルセーフ設計など)
⑤ 事故が発生した後のダメージの低減を図る

これらの処置は，番号が若いほど効果は高くなりますが，労力やコストはより多くかかると言われています．

具体的な方法を考えてみましょう．欠陥を予防するためには，ハードウェアなら故障率の低い部品を採用する，ソフトウェアなら実績のあるソフトウェア部品を再利用する，などの方法があります．欠陥を除去するには，故障を検知しやすい構造に設計するなどの方法があります(具体例として，センサ異常の検出方法を後述する)．故障として顕在化した後の対策としては，副次的影響をどう低減するか，冗長系をどう織り込んでおくかなどを検討します．

処置に関するキーワードを以下に示します．安全性設計の方策を検討するときに，これらを利用します．

① 避ける
② 検知・警報する
③ 障害や故障を除去・修正する
④ 冗長に対応する
⑤ 故障でも動作できるようにする
⑥ フェイルセーフを実施する
⑦ 監視する，要因を分離する
⑧ 分離ソフトウェア構造とする
⑨ より安全なカーネル構造とする(マイコンでいうと，スーパバイザ・モードを利用したカーネル設計など)

また，検知を織り込む場所を工程ごとに見ていくと，以下のようになります．

① ハードウェアで検知する(ハード故障や異常命令などに

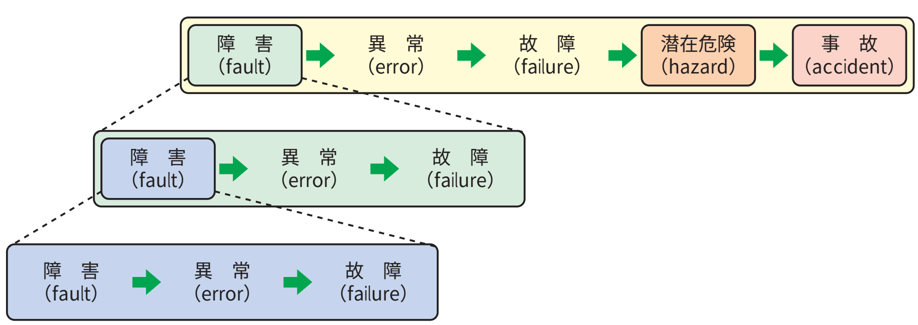


図1
障害－異常－故障
障害(fault)－異常(error)－故障(failure)の連鎖が，事故につながる．

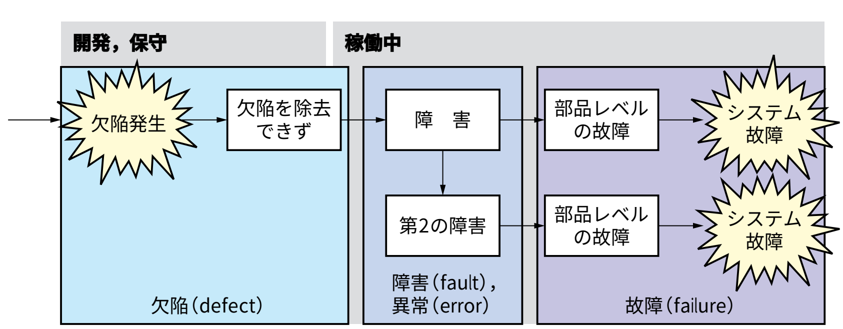


図2
欠陥－障害－故障の因果関係モデル
欠陥(defect)が発生し，それが除去できない欠陥の場合，障害が発生する．障害は異常，故障と連鎖し，部品レベルの故障がシステム故障につながる．

対する処置）

② ソフトウェアで検知する（実行エラー処置や，事前構造設計での処置）

③ プロセスを独立させて検知する（別プロセスでモニタする）

④ システムを独立させて検知する（独立したハードウェアなどを利用して検知，処置する．スーパバイザ，オブザーバなどと呼ばれる）

2．危険のもとを分析する代表的な手法を知る

障害や故障，潜在危険などを分析する代表的な手法としては，故障モード影響解析（FMEA：failure mode and effect analysis），フォールト・ツリー解析（FTA：fault tree analysis），HAZOP（hazard and operability）スタディーなどがあります（表1）．またFMEAには，製品設計段階で実施する設計FMEA（DFMEA）と製造工程設計段階で実施する工程FMEA（PFMEA）があります．

FMEAとFTAは，自動車分野における品質システム規格「QS-9000」（現在はISO 16949）にも定義されている，ポピュラーな手法です．HAZOPは，化学プラント分野などでよく使われています．

以下，FMEAとFTAについて，具体的な例を示しながら説明します．

● 故障モード影響解析（FMEA）

「故障モード」とは，故障状態の物理的/化学的な変化を表現したものです．具体的には，変形，断線，短絡，開放，折損，摩耗，ノイズ，特性の劣化などが故障モードにあたります．一方，「故障」とは，機能を満たさなくなったことを表現する言葉です．異なる概念なのですが，混同しやすいので注意が必要です[注1]．

故障モード影響解析（FMEA）は，製品の機能を構成する部品ごとに故障モードをリストアップし，これらの故障モードが製品に及ぼす影響を予想することで，潜在的な事故を設計段階で予測・抽出します（表2）．

さらに，これらの故障モードに対して故障が発生した場合の影響の大きさ（厳しさ）や，発生する確率（発生率），発生の見つけにくさ（検出率）などをそれぞれ10段階で評価し，積をとってランク付けを行います（危険優先数）．この分析結果を対策に反映し，事故や故障を予防します．

注1：故障モードは部品レベルに落としていくが，機能やサブ機能の故障モードと混在しやすい．そのため，ISO 16949では部品レベルでの故障モードのことを，「故障モードとその故障の，潜在的原因または故障メカニズム」という表現で説明している．

2

表1　危険の源を分析する手法の例

名　称	目　的	概　要	特　徴
故障モード影響解析（FMEA：failure mode and effect analysis）	故障の影響を分析する	故障の可能性があるもの（主に部品）に対して，どのような影響（故障モード）があるかをリストアップし，発生頻度・影響度・検出難易度に基づいて，優先順位を明確にしながら設計対策を実施していく	ボトムアップ的
フォールト・ツリー解析（FTA：fault tree analysis）	故障や事故の発生頻度を分析する	望ましくない事象を定義して，その事象を発生させる要因を抽出し，それぞれの発生確率を検討する	トップダウン的
HAZOP（hazard and operability）スタディー	システムに発生する可能性のある潜在危険と，運転性への障害を解析する	システムあるいはプロセスを中心として，ガイドワードを用いて解析する	トップダウンとボトムアップの中間

表2　故障モードと影響解析表

ISO 16949に定義されている故障モード影響解析（FMEA）のフォーマット例．「厳しさ」，「発生率」，「検出率」のランクの表現は製品ごとに異なるので，ランク表の運用規定を作る必要がある．また，「RPNがいくつ以上なら是正を行う」という勧告是正基準を作成し，この基準を満足しない項目について対策を検討する．この表の「対策」の右側にある項目は，対策後の数値を示している．なお，影響度については，単独でも「いくつ以上なら是正を行う」という処置を併用する．

No.	機能	故障モード	故障の影響	厳しさ	故障の原因メカニズム	発生率	現在の設計管理予防	現在の設計管理検出	検出率	危険優先数	対策	厳しさ	発生率	検出率	危険優先数
11	＊＊＊	＊＊＊	＊＊＊	6	酸化接触不良	4	＊＊＊	＊＊＊	5	120	＊＊＊	6	4	2	48
					熱ストレス破断	2	＊＊＊	＊＊＊	5	60					

● フォールト・ツリー解析（FTA）

図3に，フォールト・ツリー解析（FTA）の例を示します．この手法では，まず初めに望ましくないトップ事象（ここでは「車速信号停止」）を定義し，その事象を発生させ得る要因をたどって抽出していきます．

1次要因（ECU入力回路故障，センサ系統故障）を列挙し，さらにその要因を挙げ，さらに…，と因果関係を追究していきます．望ましくない事象を漏れなく列挙し，システムの故障を発生させる事象との因果関係を，論理記号（OR，AND）を用いてツリー状に表現します．また，事象ごとに故障率を割り当てます．そこから逆に確率を集計していき，トップ事象や上位要因の確率を求めます．

トップ事象の確率が大きい場合，確率の大きいルートに対して対策を実施します．それを，トップ事象の確率が小さくなるまで繰り返します．

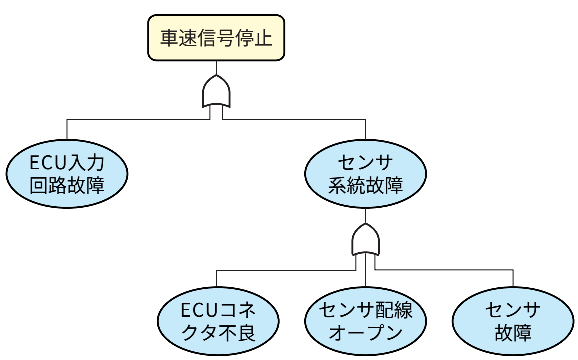


図3　フォールト・ツリー解析（FTA）
ここでは，自動変速機の制御において「速度情報がマイコンへ伝送されない」という障害に対して解析している．

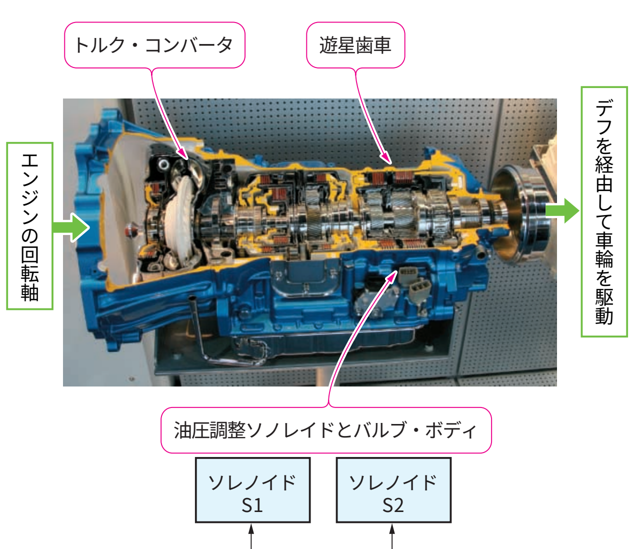


写真1　自動変速機（automatic transmission）の外観

設計段階では，FMEAを検討した後に，抽出した故障モードをさらに掘り下げるためにFTAを併用します．また，検証段階で見つかった不具合を論理的に追求し，原因を究明するためにもFTAが使われます．

3．フェイルセーフ設計で安全性を高める

安全性を高めるための一般的な設計手法の一つとして，フェイルセーフ構成の設計例を説明します．写真1に示すのは，自動車の自動変速機（automatic transmission）です．ここでは，速度センサと2個のソレノイド（アクチュエータ）を制御して，4速の変速を行うシステムを考えます．

実際の自動変速機は，遊星歯車やクラッチの制御など，内部で複雑な動作を行いますが，ここでは話を単純にするために，ソレノイドのON/OFFで油圧ON/OFFを作り出し，バルブ・ボディでそれにうまく対応するための内部変速調整回路が組み込まれていると仮定します．また，実際の自動変速機はアクセルと連動していますが，ここではそれを考慮しません．

システム構成を図4に示します．通常動作（故障のない状況）では，マイコンが速度センサからの信号を受けて計算し，ハイサイドON/OFF回路に対してソレノイドのON/OFF指示を出します．変速段に対応するソレノイド状態を表3(a)に，変速に対応する速度を表3(b)に示します．

● 故障の場合の検討

ここでは，図4のシステムに対して故障モード影響分析（FMEA）を実施してみます．具体的には，各機能ブロックに対して，順番に故障を検討していきます．例えば，速度センサの故障とは，「速度を検出し，マイコンへ伝達する」

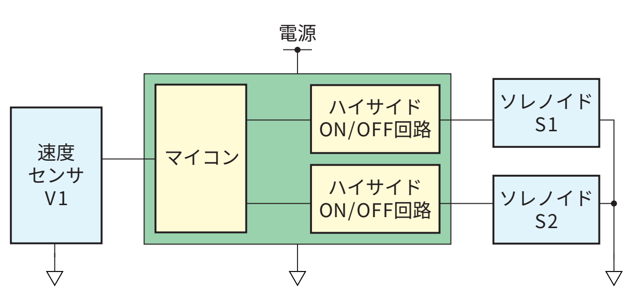


図4　自動変速機のシステム構成
ここでは説明をかんたんにするために，システムを簡略化した．

機能が欠けることになります．ここから，まず故障モードを検討し，実際にどのような故障が想定されるかを洗い出していきます．

この場合，下記の点を意識して作業を進める必要があります．

- 実際の物理的状況を明確にする
- 機能と機能を結ぶ伝送機能を含める
- 2次的影響も抽出する

なお図4では，速度センサがマイコンに直接つながっているように表現していますが，実際にはセンサ信号がセンサ・コネクタで配線につながり，その配線からECUのコネクタを経由し，プリント基板の配線を経由して入力回路へつながり，マイコン・ポートに入力されます．これは，「速度センサ」や「マイコン」といった機能ブロックの中には，さらにサブ機能があることを意味します．

故障モードは，先ほども説明したように，物理的/化学的な変化状況を表現します．各サブ機能について，具体的に分かる故障モードとして抽出します．例えば，コネクタのオープン故障の場合，故障モードとしては「接点の酸化」や「コネクタ・ピンの銅線とのカシメ緩み」があります．配線の故障なら，途中に中継ボックスがあるかどうかに注意が必要ですし，咬み込みなどによる車両ボディ（グラウンド電位）短絡も考えられます．配線のボディ短絡であれば，2次的な短絡電流の影響も検討する必要があります．

このように，さまざまな故障モードが抽出されますが，その影響は比較的まとまります．例えば，速度センサの場合は，故障の影響は「速度情報がマイコンへ伝送されないことによる変速不良」などになります．

表3　自動変速機の設定と入出力値

変速段		1速	2速	3速	4速
ソレノイド状態	S1	ON	ON	OFF	OFF
	S2	OFF	ON	ON	OFF

(a) 変速段とソレイノイド状態の対応

速度（km/h）	15	30	55
変速	1速→2速，2速→1速	2速→3速，3速→2速	3速→4速，4速→3速

(b) 変速のしきい値

● 故障モードの影響を分析する

故障モード影響分析では，影響を数値化し，重要度を具体化していきます．ここでは数値化はしませんが，どのような影響と対応するかを説明します．

前述した速度センサの例で，「速度情報がマイコンへ伝送されない」という故障を実際の制御の場面に当てはめて考えてみます．例えば，走行中に速度情報がマイコンに伝わらない場面を考えます．速度センサの値は，走行前や停止中は速度＝ゼロとなるので，一般的な速度センサ（マグネットとコイル方式）では故障と見分けがつきません．そのため，制御としてシフトダウンが行われることになります．これに対する方策の候補を以下に列挙します．

- 通常の走行で発生しない急激な速度変化については，故障の影響とみなして処置する
- 急にシフトダウンしないような機構を併用する（ワンウェイ・クラッチなど）
- センサを2系統にして処置する（冗長化）
- 走行前にセンサ故障を検知して，変速しない（センサにディジタルICを内蔵して常にシリアル通信を行う仕様とすれば，その通信が途絶えることでセンサ機能の故障を検知できる）

このようにして，ほかの機能ブロックやサブ機能も順に分析し，部品の故障モードを抽出しては影響を分析していくと，例えば以下のような処置を検討しようという考えが浮かびます．

- マイコンの故障を監視するウオッチドッグ回路と，リセット時に変速を4速側へ移行させる回路[注2]
- ソレノイドのオープン・ショートを検出する回路
- ソレノイド・ショート時のハイサイド回路保護
- ソレノイドS1がONでS2がON（2速）のとき，S2の故障時にはS1をOFF（4速）にする処置（1速に移行することを防ぐ）
- ハイサイド回路のON/OFF故障への処置
- 電源ON時に故障診断を実施する処置

● 機能を最適配置するためには設計者の連携が大切

このようにフェイルセーフを検討するとき，その処置をソフトウェアで実施するのか，機構で実施するのかなど，システム全体でトレードオフや最適配置を検討することに

注2：自動車が走行中に速度情報が0になったとき，2速から1速にシフトダウンしてしまうと自動車が低速でしか走れなくなってしまう．変速しないか，または4速側に移行することにより，その場で停止するよりも安全に走行できる．

2

なります．

IEC 61508の第3部で示されている開発ライフ・サイクル(図5)にも，このことが示されています．システム全体(E/E/PE)の安全要求仕様からソフトウェア安全要求仕様に落とし込むとき，また，ソフトウェア・アーキテクチャを検討するときに，システム全体のアーキテクチャとソフトウェアのアーキテクチャを調整しながら設計するという流れになっています．

これはシステム設計において重要な点です．実現しやすい個所で処置を分担できれば，システムをより簡素なものにできます．そのためには，電子回路の設計技術者と機構設計技術者，ソフトウェア設計者とハードウェア設計者が連携することが大切です．

また，妥当性確認テストからソフトウェア安全要求仕様に対して矢印があるのは，今まで見てきたように，故障モード解析などを行った結果，どのように制御するのかを安全要求仕様にフィードバックしながら仕様を作り込んでいく必要があることを表しています．

● ISO 9000やCMMへの取り組みが機能安全につながる

現在，ISO 9000やCMM(Capability Maturity Model；能力成熟度モデル)に取り組んでいる企業の方も多いと思います．機能安全も「品質や安全は，良いシステム(きまり)やプロセスによって作り込まれる」と考える点はISO 9000などと共通しています．機能安全の多くの取り組みや手法は，CMMでいうプラクティス間でマッピングできると言われています．

従って，ISO 9000[注3]やCMM[注4]に着実に取り組み，力をつけることが，その延長として機能安全への準備が進むことでもあります．

4. 自動車制御の安全を確保するための技術動向

車載システムは安全面に特に配慮しながら開発されてきましたが，車載システムがますます大規模に，複雑になるに従って，安全を確保するための新しい技術が求められています．ここでは，リアルタイムOSに求められる保護機能と，最近注目されている形式検証について紹介します．

● リアルタイムOSの保護機能に期待

最近では車両制御が大変複雑になってきています．さまざまな車両機能をまたいだ制御が，一つのコンピュータの中に組み込まれて，実行されるようになってきています(統合制御と呼ばれる)．これらの機能をリアルタイムOSが統括しますが，各機能を独立させ，信頼性を高めるには，リアルタイムOSの保護機能が重要です．

保護機能としては，各機能が利用するメモリ領域を保護するメモリ保護機能のほかに，タスクの処理時間オーバを監視する時間保護機能も重要になってきており，これらを組み込んだプラットホームが標準化されることを期待しています．

注3：自動車分野では，米国ビッグ・スリーの要求事項を上乗せした品質システム規格「QS-9000」，さらに現在は，国際標準化されたISO/TS 16949などがある．

注4：最近は，CMMI(Capability Maturity Model Integration)やISO 15504，SPICE(Software Process Improvement Capability dEtermination)，今後の自動車分野ではAutomotive SPICEなどがある．

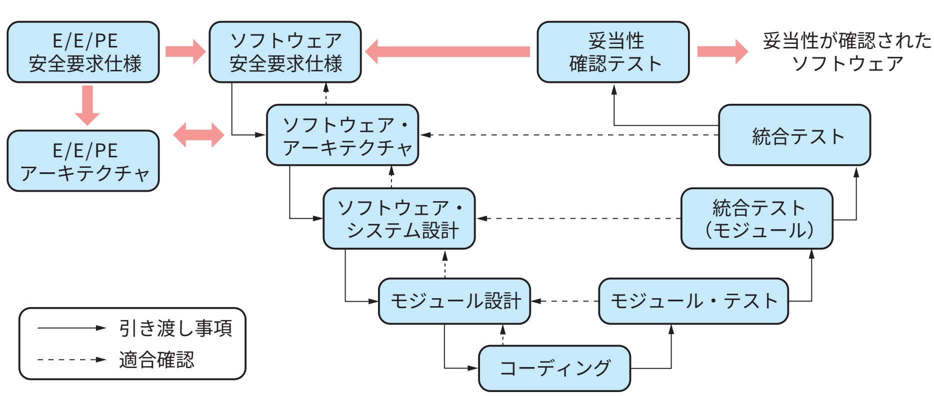


図5
IEC 61508 第3部で示されているソフトウェアの開発ライフ・サイクル(V字モデル)
「E/E/PEアーキテクチャ」と「ソフトウェア・アーキテクチャ」をつなぐ双方向の矢印は，ここで設計のトレードオフを検討するべきであることを意味している．また，「妥当性確認テスト」から「ソフトウェア安全要求仕様」への矢印は，各種の分析や確認した結果を安全要求仕様にフィードバックし，仕様を作り込んでいく必要があることを示している．

● 形式検証とモデリングのトレンド

IEC 61508で推奨している手法として，形式検証(formal method)がしばしば話題になります．現在，形式検証技術が進んでいるのは，主として離散事象(discrete event；非同期的で不連続なタイミングで生起するイベント)を対象としたモデル(状態遷移図など)についてです．

自動車などは通常，連続事象と離散事象を両方扱う必要があります(これをハイブリッド・モデルという)．また「実行可能な仕様書」などと呼ばれるような，システム上流からの動作検証などを行うには，電子回路のほかに電磁気や油圧などを総合的に扱えるヘテロジニアスな要素も合わせてモデリングできる必要があります[注5]．

「ソフトウェア実行構造がタイム・トリガ方式の同期構造であることを前提とする」などと限定すれば，フランスEsterel Technologies社のモデリング＆形式検証ツール「SCADE」などがありますが，ハイブリッド・モデルの形式検証はまだ発展途上です．

また，別のアプローチとして，以下のような取り組みが行われています．

1）C言語で形式検証する

C言語は形式言語ではありません．しかし，フォーマル・ツールを適用することで，ある程度の形式検証は可能です．例えば，抽象解釈理論を適用して開発されたフランスPolyspace Technologies社のエラー検出ツール「Polyspace Verifier」は，ソース・コードから，すべての入力変数範囲においてオーバフローやデッド・コードがないかどうかを検証します．また，最悪実行処理時間(WCET；worst case execution time)も，抽象解釈理論を応用したツールで解析できます．

2）UML＋ADLで形式検証する

現在，VDMなどの形式検証言語で仕様から記述する，などという方法が話題になっていますが，なじみのない表記を理解・習得する必要があります．一方，フローチャートに似た形式言語で，国際電気通信連合(ITU：International Telecommunication Union)が推奨するSDL(Specification and Description Language)があります．これを使えば，テスト・ケース自動生成などで有効に検証できます．SDLは，FlexRay通信仕様書の動作記述にも利用されています．

また，UML2.0は完全な形式言語ではありませんが，拡張プロファイルでモデリング定義すれば，形式言語として利用できます．

ただし，形式化したモデルでC言語レベルの振る舞いまでをすべて記述しようとすると，VDMと同じように，習得するのが大変です．むしろ，アーキテクチャ・レベル記述に絞って形式化して利用すれば，その分容易に利便性を得られます．

最近の流れとしては，アーキテクチャ記述言語(ADL：Architecture Description Language)をUML拡張プロファイルで定義し，システムやソフトウェアのアーキテクチャ部を記述して評価するという方法が，モデリング表記の分かりやすさもあって，盛んに取り組まれています．これは，IEC 61508が，アーキテクチャ設計工程とソフトウェア・モジュール設計工程を明確に区別し，アプローチを規定していることにも関係しています．

注5：このため，ハイブリッド・モデルの記述が容易な米国The MathWorks社の数値計算/信号処理設計ソフトウェア「MATLAB」などがよく利用されている．

2

参考・引用*文献

(1) 中嶋洋介；安全とリスクのおはなし，日本規格協会，2006年6月．
(2) 小野寺勝重；FMEA手法と実践事例，日科技連出版社，2006年8月．
(3) 岸 知二，野田夏子，深澤良彰；ソフトウェアアーキテクチャ，共立出版，2005年6月．
(4) 鈴木和幸，田中健次ほか；正しい故障・信頼性解析で事故を防げ！ トラブルを回避せよ！，機械設計，2005年10月号，日刊工業新聞社．
(5) Prof. Dr. Stefan Kowalewski；Analytische und konstruktive Methoden für sichere Software, GMA Workshop, Sep. 2004.
(6) 3rd World Congress for Software Quality(第3回 世界ソフトウェア品質会議)のホームページ，http://www.isqi.org/isqi/eng/conf/wcsq/3/

すずむら・のぶやす
アイシン精機(株)

<筆者プロフィール>
鈴村延保．1977年，広島大学工学部を卒業．アイシン精機にて，自動変速機やアクティブ・サスペンション，ABS，ESC用のコンピュータとそれに関連するICの開発・製品化に従事．